AMENDMENTS TO THE CLAIMS

Please amend the claims as indicated hereafter.

1. (Currently Amended) A method of data object transformation between a middleware and a application, the method comprising:

receiving a message from a messaging middleware by a data transformation adapter, the message including one or more data objects in an eXtensible Markup Language (XML) ~~of a first object type~~, wherein the message is a first communications format;

converting by the data transformation adapter the message from the first communications format to a second communications format;

converting by the data transformation adapter the one or more data objects in XML to a non-eXtensible Markup Language (non-XML) ~~from the first object type to a second object type~~, wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~, each of the one or more transformation classes generated using mapping rules, the mapping rules including ~~eXtensible Markup Language (XML)~~ XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~; and

transmitting by the data transformation adapter the one or more ~~second object type~~ data objects in non-XML to an application.

2. (Previously Presented) The method according to claim 1, wherein the first communications format includes a middleware-dependent format, and the second communications format includes a middleware-independent format.

3. (Previously Presented) The method according to claim 1, wherein each of the one or more data objects includes a Java object.

4. (Currently Amended) The method according to claim 1, wherein the ~~first object type~~ XML includes a domain object model type and the ~~second object type~~ non-XML includes an application-specific object model type.

5. (Previously Presented) The method according to claim 1, further comprising: registering the application with the messaging middleware; and transmitting high-level function calls to the application.

6. (Currently Amended) The method according to claim 1, the method further comprising:

receiving a second message from the application, the second message including one or more data objects in non-XML ~~of the second object type~~;

converting the one or more data objects in non-XML to XML ~~from the second object type to the first object type~~, wherein the one or more data objects are converted using a second set of one or more of the transformation classes;

generating a communications line dependent message, the communications line dependent message including the one or more ~~first object type~~ data objects in XML; and

transmitting the communications line dependent message to the messaging middleware.

7. (Canceled)

8. (Canceled)

9. (Canceled)

10. (Canceled)

11. (Currently Amended) A data transformation adapter having program instructions stored in memory, the program instructions comprising:

generating a first object model and a second object model, the first object model including a plurality of data objects in an eXtensible Markup Language (XML) ~~of a first object type~~, and the second object model including a plurality of data objects in a non-eXtensible Markup Language (non-XML) ~~of a second object type~~;

storing the first and second object models in one or more memories;

generating mapping rules, the mapping rules including ~~eXtensible Markup Language (XML)~~ XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~;

generating a plurality of transformation classes using the first and second object models and the transformation mapping rules, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~;

receiving one or more data objects;

converting the received one or more data objects, via the transformation classes, ~~from~~ (1) in XML to non-XML ~~the first object type to the second object type~~; or (2) in non-XML to XML ~~from the second object type to the first object type~~ ; and

transmitting the converted one or more data objects.

12. (Previously Presented) The method according to claim 11, wherein each of the one or more data objects includes a Java object.

13. (Currently Amended) The method according to claim 11, wherein the ~~first object type~~ XML includes a domain object model type and the ~~second object type~~ non-XML includes an application-specific object model type.

14. (Cancelled)

15. (Previously Presented) The method according to claim 11, wherein the one or more data objects are received from a messaging middleware.

16. (Previously Presented) The method according to claim 11, wherein the one or more data objects are received from an application, the application being coupled to a messaging middleware.

17. (Currently Amended) A system for data object transformation, the system comprising:

one or more processors;

one or more memories coupled to the one or more processors; and

a data transformation adapter having program instructions stored in the one or more memories, the one or more processors being operable to execute the program instructions, the program instructions including:

receiving a message from a messaging middleware, the message including one or more data objects in an eXtensible Markup Language (XML) ~~of a first object type~~, wherein the message is in a first communications format;

converting the message from the first communications format to a second communications format;

converting the one or more data objects in XML to a non-eXtensible Markup Language (non-XML) ~~from the first object type to a second object type~~, wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~, each of the one or more transformation classes generated using mapping rules, the mapping rules including ~~eXtensible Markup Language (XML)~~ XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~; and

transmitting the one or more ~~second object type~~ data objects in non-XML to an application.

18. (Previously Presented) The system according to claim 17, wherein first communications format includes a middleware-dependent format, and the second communications format includes a middleware-independent format.

19. (Previously Presented) The system according to claim 17, wherein each of the one or more data objects includes a Java object.

20. (Currently Amended) The system according to claim 17, wherein ~~first object type~~ XML includes a domain object model type and the ~~second object type~~ non-XML includes an application-specific object model type.

21. (Currently Amended) The system according to claim 17, wherein the program instructions further include:

receiving a second message from the application, the second message including one or more data objects in non-XML ~~of the second object type~~;

converting the one or more data objects in non-XML to XML ~~from the second object type to the first object type~~, wherein the one or more data objects are converted using a second set of one or more of the transformation classes;

generating a communications line dependent message, the communications line dependent message including the one or more ~~first object type~~ data objects in XML; and

transmitting the communications line dependent message to the messaging middleware.

22. (Currently Amended) A system for data object transformation, the system comprising:

a communications line;

a transformation adapter coupled to the communications line, the transformation adapter including:

an assembly/disassembly layer configured to convert messages from a first communications format to a second communications format;

a transformation layer configured to convert data objects in an eXtensible Markup Language (XML) to a non-eXtensible Markup Language (non-XML) fro~~m a first object type to a second object type~~ using one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~; and

a method invocation layer;

a transformation class generator coupled to the transformation adapter, the transformation class generator configured to generate the one or more transformation classes using transformation mapping rules, the mapping rules including ~~eXtensible Markup Language~~

~~(XML)~~ XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~; and

an application coupled to the transformation adapter, wherein the application transmits data to and receives data from the method invocation layer.

23. (Previously Presented) The system according to claim 22, wherein the communications line includes messaging middleware.

24. (Previously Presented) The system according to claim 22, wherein each of the one or more data objects includes a Java object.

25. (Currently Amended) The system according to claim 22, wherein the ~~first object type~~ XML includes a domain object model type and the ~~second type~~ non-XML includes an application-specific object model type.

26. (Currently Amended) An apparatus for data object transformation, the apparatus comprising:

means for generating a first object model and a second object model, the first object model including a plurality of data objects in an eXtensible Markup Language (XML) ~~of a first object type~~, and the second object model including a plurality of data objects in a non-eXtensible Markup Language (non-XML) ~~of a second object type~~;

means for storing the first and second object models;

means for generating transformation mapping rules, the mapping rules including ~~eXtensible Markup Language (XML)~~ XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~;

means for generating a plurality of transformation classes using the first and second object models and the transformation mapping rules, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML ~~from the first object type to the second object type~~;

means for receiving one or more data objects;

means for converting the received data objects, via the transformation classes, in XML to non-XML ~~from the first object type to the second object type~~; and

means for transmitting the converted one or more data objects.